```
program wsmaker(input,output);

const
   maxsize = 17;
   fillch = '-';
   curch = '*';
   up = 0;
   upright = 1;
   right = 2;
   downright = 3;
   down = 4;
   downleft = 5;
   left = 6;
   upleft = 7;

type
   gridtype = array [1..maxsize,1..maxsize] of char;
   ptrwordtype = ^wordtype;
   wordtype = record
                 word: packed array[1..maxsize] of char;
                 wordsize: 1..maxsize;
                 nextword,
                 prevword: ptrwordtype;
              end;
   direction = up..upleft;

var
   wordfile,
   gridfile: text;
   outfile,
   infile: packed array[1..12] of char;
   grid: gridtype;
   maxrow,
   maxcol,
   currow,
   curcol: 1..maxsize;
   aword,
   firstword,
   lastword,
   wordsleft,
   wordsused,
   firstused: ptrwordtype;
   i,
   seed: integer;

function random: integer;

begin
   seed := (131071 * seed) mod 2147483647;
   random := seed div 2147484;
end; (* random *)

procedure initialize;

var i,j: integer;

   procedure readwords;

   (* words are assumed to be in the file separated by commas *)

   var

      ch: char;
```

```
var

   ch: char;
   len: 1..maxsize;

   procedure readaword;

   begin
      len := 0;
      while ((ch <> ',') and (ch <> '.')) and (not eof(wordfile)) do
         begin
            len := len + 1;
            wordsleft^.word[len] := ch;
            read(wordfile,ch);
         end;
      wordsleft^.wordsize := len;
      writeln('word is ',wordsleft^.word,'  length is ',wordsleft^.wordsize);
   end; (* readaword *)

begin (* readwords *)
   new (firstword);
   firstword^.wordsize := 0;
   lastword := firstword;
   open(wordfile,infile,old);
   reset(wordfile);
   if eof(wordfile)
      then writeln('empty file')
      else begin
              read(wordfile,ch);
              repeat
                 new(wordsleft);
                 readaword;
                 with wordsleft^ do
                    begin
                       if firstword^.wordsize < wordsize
                          then
                             begin
                                nextword := firstword^.nextword;
                 if nextword <> nil then nextword^.prevword := wordsleft;
                                firstword := wordsleft;
                             end
                          else
                             begin
                                aword := firstword;
                                repeat
                                   aword :=aword^.nextword;
                                until aword^.wordsize < wordsize;
                                prevword := aword^.prevword;
                                aword^.prevword := wordsleft;
                                prevword^.nextword := wordsleft;
                                nextword := aword;
                             end (* else *)
                    end; (* with *)
                 read(wordfile, ch);
              until eof(wordfile);
           end;
      close(wordfile);
   end; (* readwords *)

   procedure orderbylength;

   begin (* orderbylength *)
   end; (* orderbylength *)

begin (* initialize *)


   write('enter seed:  ');
```

(* make list big to small *)

```
if wordsize > firstword↑.wordsize
   then begin
      nextword := firstword;
      firstword↑.prevword := wordsleft;
      firstword := wordsleft;
   end
else begin
   aword := firstword;
   while (wordsize <= aword↑.wordsize) & (aword↑.nextword <> nil) do
      aword := aword↑.nextword;
   if wordsize > aword↑.wordsize
      then begin
         nextword := aword↑.nextword;
         aword↑.prevword := wordsleft;
         prevword := aword↑.prevword;
      end
```

```
begin (* initialize *)


   write('enter seed:  ');
   read(seed);writeln;
   write('Enter number of rows:  ');
   read(maxrow);
   writeln;
   write('Enter number of columns:  ');
   read(maxcol);
   for i := 1 to maxrow do
      for j := 1 to maxcol do
         grid[i,j] := fillch;
   write('Enter words file name:  ');
   readln;readln(infile);
   writeln;write('Enter output file name:  ');
   readln(outfile);writeln;
   readwords;
   firstused := nil;
   orderbylength;
end; (* initialize *)

procedure makeit;

var
   dir: direction;


   procedure pickaspot;

   begin (* pickaspot *)
      repeat
         currow := 1 + random mod maxrow;
         curcol := 1 + random mod maxcol;
      until grid[currow, curcol] = fillch;
   end; (* pickaspot *)

   function pickadirection: direction;

   begin (* pickadirection *)
      pickadirection := random mod upleft;
   end; (* pickadirection *)

begin (* makeit *)
   pickaspot;
   dir := pickadirection;
end; (* makeit *)

procedure printit;

var i,j: integer;

begin
   open(gridfile,outfile);
   rewrite(gridfile);
   for i := 1 to maxrow do
      begin
      for j := 1 to maxcol do
         begin
            write(grid[i,j]:3);
            write(gridfile,grid[i,j]:3);
         end;
      writeln;writeln(gridfile);
      end;
   writeln(gridfile);
   writeln(gridfile);


   writeln(gridfile,'WORDS USED:');
```

```
    writeln(gridfile);
    writeln(gridfile);


    writeln(gridfile,'WORDS USED:');
    aword := firstused;
    while aword <> nil do
       begin
          writeln(gridfile,aword^.word);
          aword := aword^.nextword;
       end;
    writeln(gridfile);
    writeln(gridfile);
    writeln(gridfile,'WORDS NOT USED:');
    aword := firstword;
    while aword <> nil do
       begin
          writeln(gridfile,aword^.word);
          aword := aword^.nextword;
       end;
    close(gridfile);
end;


begin
   initialize;
   makeit;
   printit;
end.
```

```
program wsmaker(input,output);
```